The Progress Electronic Magazine

In this issue:

Publisher's Statement: 2
Coding Article: Poor Man's Data Server 4
Problem and setting: 4
Solving the problem: 4
Solution Mechanics: 5
Implementation Steps: 7
The ASP program 8
Progress Test Procedure 9
Sample output of the script 10
Conclusions and additional ideas: 11
Management Article: Web Services 12
Community Announcements: Error! Bookmark not defined.
Product Announcements: 14
Service Express 14
Survey Software 14
Publishing Information: 15
Article Submission Information: 15

Did you sign up to receive this E-Zine? Send email to sauge@amduus.com to subscribe or fill out the forms at http://www.amduus.com/online/dev/ezine/EZineHome.html ! It's free! (Though donations are certainly welcome – whatever you feel is fair!)

Publisher's Statement:

Often in the Progress world we bump into the integration needs with disparate technologies. We need to pull data out of the system, put data into an application, or inform the system that we are doing something. It's not a perfect world of pure Progress applications! :P

In this E-Zine, is an article of interest from Steven Jellin that allows a UNIX computer to pull in data from an application on a Windows based computer. It is simple, works, and is illustrative of a strategy to integrate applications through web interfaces – indeed the makings of a "web service."

To that end, I will also speak a little bit about the gotcha's one can expect in web services in the management article. This little buzzword is getting a lot of press lately, and while I do not want to downplay the idea, it is time to interject some reality into the whole thing.

To your success,

Scott Auge
Founder, Amduus Information Works, Inc.
sauge@amduus.com

Amduus Information Works, Inc.
http://www.amduus.com
scott_auge@yahoo.com sauge@amduus.com

Creation of modules and products for re-sale
as well customized Internet/Intranet programming
for E-Business in the marketing/manufacturing/
service and law enforcement industries.

Reach over 700 programmers and companies.

Your ad could be here!

Advertise in the E-Zine for $10.00 per issue!

Webspeed a little pricy for you?

Contact Amduus Information Works, Inc. about our ASP services.

Develop and Deploy you applications on our servers in our datacenter or yours.

sauge@amduus.com

Coding Article: Poor Man’s Data Server

(Or a really cheap way to get data from an ODBC compliant database).

Written by Steven L. Jellin

Problem and setting:

You need to get information from some Ms Sql (or some ODBC compliant server)
Don’t have the money to buy a Dataserver.

Your particular progress configuration / version doesn’t support the relevant Dataserver.

Free Apps available at Amduus

CMS - Web Content Management System

Blue Diamond - Webspeed like transaction server

In the case that prompted this solution to be developed, the relevant dataserver existed, but was not available for V9.1A. There was also no time to wait for, and install, and test a move to V9.1C. In the current situation, all that was needed was to be able to read data from the relevant Sql Server and process it.

The environment consists of CHUI applications running on *nix machines. We were perhaps fortunate in that we have a Win 2k server that could be used as an intermediary between our *nix boxes and the target SQL server sitting elsewhere.

Solving the problem:

As mentioned above, we have our CHUI application sitting on a *nix machine that needs to talk to an Ms Sql server sitting elsewhere. We also have a Win 2k server running IIS5, sitting behind a firewall.

The immediate and quickest solution that I was able to think of involved using an ASP page on the web server to do the communication with the Sql Server (with passed parameters). This is all very well and fine, but the problem of getting the data back into the CHUI application also needed to be addressed.

At the time of initial development, we did not have access to a V9 environment or an assurance that we would be sitting on V9 (we where on V8) by the deadline. Hence sockets where out of the question. Additionally the method used in the end, allows for the use of SSL at no extra programming overhead.

Fortunately the progress 4GL provides the needed mechanism for getting the data into progress. The input through some-os-command which when coupled with lynx provided the answer.

At this point, I should mention that other smaller / faster alternatives to lynx exist. These include wget and curl amongst others. Our problem was that I couldn't compile them under our variant of Unix (out dated gcc, no gmake etce etc).

Solution Mechanics:

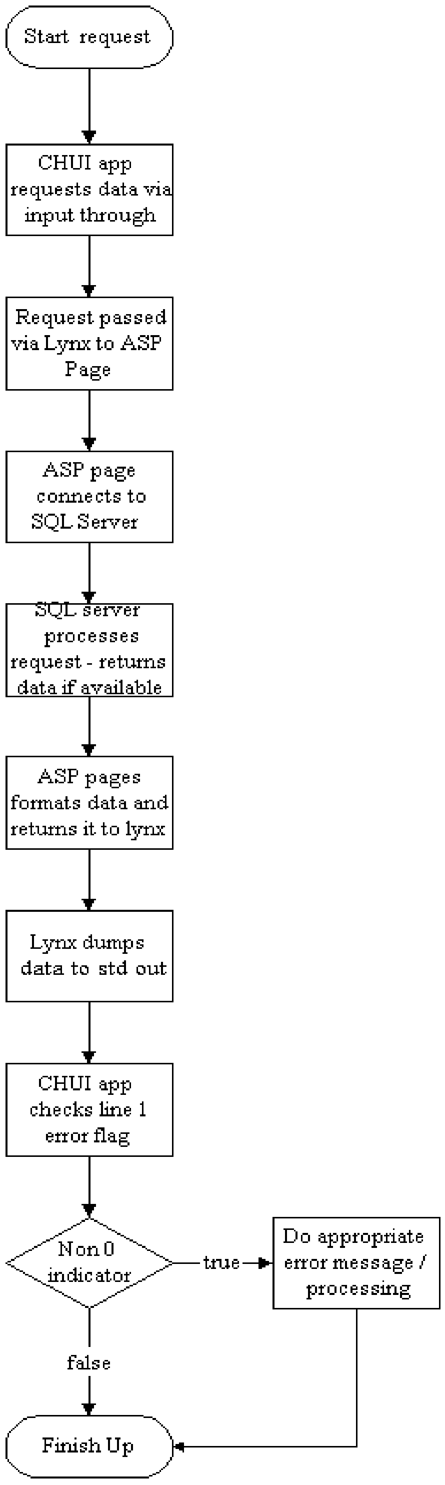
Start request
CHUI app requests data via input through
Request passed via Lynx to ASP Page
ASP page connects to SQL Server
SQL server processes request - returns data if available
ASP pages formats data and returns it to lynx
Lynx dumps data to std out
CHUI app checks line 1 error flag
Non 0 indicator
true
Do appropriate error message / processing
false
Finish Up

Figure 1 Simplied flow diagram (skip error checking)

How you process the incoming data is entirely up to you, my approach is to load it all into a temp table, and then process according to the applications needs. At all stages if errors occur, an error number is returned to show where the error occurred. If lynx itself fails, then the first line found by the CHUI application is non zero anyway (generally a unix error, or lynx complaining that it can't connect).

Before being able to use the ASP script and sample progress file some prerequisites need to be sorted out.

Implementation Steps:

1. If you haven't already done so, install IIS on the NT machine (or PWS on a 9x machine etc).
2. Set up an ODBC connection (from control panel etc). Set up should be straight forward and is thus left out of this discussion. For test purposes if you have access to the Northwind database you can set up a DSN for that too.
3. Decide where you want your ASP script to reside, e.g. <webroot>\odbcasp\ . Ensure that the relevant directory has execute permissions, else the script won't run. Again the exact particulars of this are beyond the scope of this document.
4. Copy the asp script into the directory.
5. Test that the script works … see the example progress code for the URL passed and substitute your DSN name. Opening up internet explorer and giving it that URL should return some data.
6. I chose to restrict access to the directory to the Unix machines and by default deny all other hosts as an added precaution.
7. Set up SSL as needed (note your lynx binary / whatever binary you use will need support compiled in – this can be problematic if your Unix box doesn't have all the needed toys).
8. Curse the author for leaving something out that you needed to do, and thank your stars that you can configure IIS and ODBC blindfolded.

Next is the code that does all the hard work, followed by the sample progress application.

Server side code that does the actual query.
Parameters:
SQL – Sql command to execute
DSN – Your system / File DSN containing connection information
UID – Userid to connect as

PWD – Password for User (optional)
DSNTYPE – Type of DSN used. For file DSN use “FILEDSN” – blank = system. (optional)

The ASP program

```
<%@ Language=VBScript %>
<%
   On Error Resume Next
   sub WriteError(ErrorString)
      response.write ErrorString & chr(10)
      If ErrorString <> 0 THEN
         Response.End
      END IF
   end sub

   '<!--Copyright S.L. Jellin 2001-->
   Sql  = unescape(request.QueryString("Sql"))
   DSN  = request.QueryString("DSN")
   UID  = request.QueryString("UID")
   PWD  = request.QueryString("PWD")
   DSNTYPE = Request.QueryString("DSNTYPE")
   IF DSNTYPE="" THEN
      DSNTYPE="DSN"   ' default
   end if
   if Sql = "" OR DSN="" OR UID="" then 'OR PWD="" then
      WriteError 1
   else
         Set con = Server.CreateObject("ADODB.Connection")
         con.open DSNTYPE&"="&DSN,UID,PWD
         If Err.number <> 0 then
            WriteError 2
         End If
         Set rst = con.execute (sql)
         If Err.number <> 0 then
            WriteError Err.Description
         End If
         'stop processing if not a query
         IF instr(UCASE(Sql),"SELECT ") = 0 then
             WriteError 0
           Response.end
         end if
         If not (rst.eof AND rst.bof) then
            SqlArray = rst.GetRows()
            If Err.number <> 0 then
                  WriteError 5
              End If
              WriteError 0
            For Row = 0 TO UBOUND(SqlArray,2) ' extent 2 = rows
                For Col = 0 TO UBOUND(SqlArray,1) ' extent 1 = Columns
/ field.value
                          'handle date : time style fields and split
into date && time
                          If typeName(SqlArray(col,row)) = "Date" then
                            Response.Write
FormatDateTime(SqlArray(col,row),2) & "," &
FormatDateTime(SqlArray(col,row),4)
                          else
                              Response.Write SqlArray(col,row)
                          end if
                          If Col <> UBOUND(SqlArray,1) then
                             response.write ","
                          end if
                Next
                response.write CHRB(10)
            Next
         else
            WriteError 6
         con.close
         set con = Nothing
         rst.close
         set rst = Nothing
         end if
   end if

%>
```

A casual glance at the ASP script indicates that date fields are split into corresponding date and time fields. (It worked for me, you might need to fiddle). Also note that I couldn't figure out if there was an easy way to dump each row as a comma separated list.
The result is some funny manipulation of an array to dump out the fields. VERY important, make sure that you have the NT machines short Date format in DMY or MDY etc. If your progress session has a different date format, just do SESSION:DATE-FORMAT = "DMY" etc before and after the imports.

Progress Test Procedure

```
/*  testfile.p */
```

```
def var lynxstr as char no-undo.
def var stat    as char no-undo.
def temp-table customer
    field cid   as char
    field cname as char.

assign lynxstr = "'http://<yourip>/odbcasp/getdetails.asp?"
       lynxstr = lynxstr + "DSN=northwind&UID=feedme&PWD=seemore&"
       lynxstr = lynxstr + "SQL=select customerid,companyname from
customers'".

ASSIGN lynxstr = REPLACE(lynxstr," " ,"%20")
       lynxstr = REPLACE(lynxstr,",","%2C").

input through VALUE("/usr/local/bin/lynx " + lynxstr + " -source").
import stat.
IF stat <> "0" THEN DO:
   MESSAGE "Captain She's gonna blow !"
   QUIT.
END.
REPEAT:
    CREATE customer.
    import customer.
end.
input close.

for each customer :
    disp cid (count) cname format "x(15)".
end.
```

It is rather important that you not miss out the single ‘ at the beginning and end of the URL (as you soon find out when leaving it out). Also very important is to tell lynx that you want –source, that is to say the exact output as it came from the web server without intermediate formatting. Not having any code handy (at the time) to escape the URL, I added code to replace space and commas which so far is all I have needed.

In case you missed it, you also need to substitute your target IIS server address for <yourip>.

Sample output of the script

```
0
ALFKI,Alfreds Futterkiste
ANATR,Ana Trujillo Emparedados y helados
```

```
ANTON,Antonio Moreno Taquería
AROUT,Around the Horn
BERGS,Berglunds snabbköp
BLAUS,Blauer See Delikatessen
BLONP,Blondesddsl père et fils
BOLID,Bólido Comidas preparadas
BONAP,Bon app'
BOTTM,Bottom-Dollar Markets
BSBEV,B's Beverages
CACTU,Cactus Comidas para llevar
CENTC,Centro comercial Moctezuma
```

Note that the first line contains a 0 which indicates that all is well.

Conclusions and additional ideas:

Create your DSN entry for your ODBC connection, setup IIS if you need to and copy the ASP script file across. Test to ensure that all is working and then proceed as normal. (Pretend your reading form a text file in progress thereafter.)

Looking at the ASP code, one will notice that it is possible to pass any SQL command across, although only SELECT is fully catered for in the above code. When I get a chance I might add parameter passing too.

I see no reason why a similar script can’t be done in PHP or as a CGI script or even as a daemon running via inetd on a dedicated port. In the case of a standalone application on a dedicated port, obviously lynx would probably not be the way to go.

For those with the patience, a better solution would probably include XML string / file passing. This would definitely be a better idea than multiple calls if you decided to use this code for multiple UPDATE / CREATE commands. (I have no intention of allowing multiple SQL strings being passed at this time, or catering for transactions at present).

There are naturally limitations including potential speed issues (although my test case above, and the use for which this was implemented have acceptable speeds). Security is also a concern, however this can be largely alleviated using SSL, or machines that are on a private subnet of the organisation (e.g. servers all connected on own subnet, with clients coming in on a different interface (ie 2 NICs), specify the private IP address of the web server and security is slightly increased). From testing it appears that the largest delay is from the initial connection to the web server, thereafter the query (in my test cases) seems to have an acceptable speed.

I hope you find this code useful, even if only as a jumping point for your own solutions. Any comment and or additions are welcome.

[Author addendum: Last minute Change
For that url I added in the about author section (end of doc), plz make sure it
http://shrike.ru.ac.za/chronicle
plus add User = guest, password = <blank> .. a guest user
user = admin, pasword = admin ... a user that can update.]

About the author: The author completed his Honours in information systems and achieved his BCom Honours at the beginning of 2002. He has just over 3 years CHUI progress experience and ASP experience in the form of an honours level system development project written entirely in ASP (the project is currently still available at http://shrike.ru.ac.za/chronicle) (He is quite impressed that the damn thing worked).

The author is currently employed as a programmer at ELCB Information Services (http://www.elcb.co.za) and may be contacted at sjellin@elcb.co.za

Management Article: Web Services

Written by Scott Auge sauge@amduus.com

There has been a lot of talk and hope about Web Services. These are actions that can be pulled together via HTTP requests. Basically it is the idea of a set of components that one can string together to make a complete application.

One service might focus on parts information – adding a part, looking up a part, deleting a part, etc. In fact, lets look a web service for parts, and where some of the gotcha's might occur.

Gotcha 1: Different encapsulation methods for the data

HTML, XML, Plain Text – a web service is not made for a browser, but to be called via an HTTP message from another system. That means, the output of the web service is not necessarily HTML for a browser. It could be XML that is parsed or even plain text that is chopped up according to rules.

In fact, even if the data encapsulation method is agreed upon, such as XML as it is quite popular right now, it may not be of the same form one expects. Sure, the two systems may be XML compatible, but are the elements stored in the XML the same name and attribute?

How about the same semantic idea of the data? This leads to Gotcha number two.

Gotcha 2: Different meanings to the data

How does one define something in the data representation? A parts location could be a single bin name, or a combination of bin name and shelf… and shelf unit… and floor… and warehouse!

A web service may be expecting more or less data elements than your system holds and is hoping to make use of. There may still be limits on the length of a part number, or limits on the attributes of a part – what if there are "kit numbers" – ways to group parts together? Or a unit of measure that is unexpected, such as cubic feet for gaseous materials?

Sure, data is symbols, and these can really mean anything. But once one starts linking up the symbols, these linkages become just as important as the symbols themselves. It is in these linkages that most knowledge is encapsulated about the entity being represented. If your linkages within your company do not meet the same expectations of linkages in the web service – you may be in for a bit of trouble.

Gotcha 3: Programs are business models and business models are opinion

Most programs are business models. Business models are generally learned by a company as that which works for the company. Business models are composed of not only the data the company needs to succeed in the marketplace, but also the decisions that are performed on that data.

What if the web service does not take into account these things? What if the web service cannot make the decisions that are desired of it? That it does not even have an understanding of the criteria much less the ability to make the decision?

Gotcha 4: Nickels and Dimes

Another aspect, is that if one rents web services (such as .net) – one may find themselves being nickel and dime'ed in unexpected ways. Looking at the costs of the service and the usage costs should be taken into account carefully – especially if one purchases the service with it's back end costs.

I do not mean to put a scare into people about web services. Some will undoubtedly decide they are the silver bullet for their problems, and yet we all know there is no such thing as a silver bullet. Software is processes, measurable attributes, and decisions – and these things change as the organization grows and learns from it's mistakes. Yet outside tools for web services may want a more stable unchanging way of operation that impedes that success for the company.

Web services could certainly perform some functions – perhaps looking up a zip code, or for the more adventurous, handling taxes for various communities, states, and federal governments. I could even set up a web service at Amduus for the sending of email – no need for exchange or sendmail expertise – merely call my program with the correct parameters and email is sent out. Web services do have a place.

Tread carefully.

About the author: Scott Auge is the founder of Amduus Information Works, Inc. He has been programming in the Progress environment since 1994. His works have included E-Business initiatives and focuses on web applications on UNIX platforms.
sauge@amduus.com

Product Announcements:

Service Express

Service Express is available for sale. It provides a web based help desk system that can be used on an intranet or the internet. Highly configurable and supports workflow. We can run it ASP under your domain or our domain. Source code is available. It ships with source code. Service Express can be purchased for as little as $4,500.

Survey Software

Amduus Information Works, Inc. is creating survey software. This software can be used on a web site to query a population of people about their views and needs. The population could be internal to a company or external to yield a better understanding of the marketplace. Documentation for the application will be available at http://www.amduus.com for free download.

The software ships with source code for better adaptability to your company's application landscape and needs.

Customers and resellers are welcome to contact Scott Auge at sauge@amduus.com for more information. Survey Express can be had for as little as $1,000.

The software can be rented out at $100.00 per survey per week of taking results.

Publishing Information:

Scott Auge publishes this document. I can be reached at scott_auge@yahoo.com.

Currently there are nearly 600 subscribers and companies that receive this mailing! This mailing is not sent unsolicited, so it is not SPAM.

Amduus Information Works, Inc. assists in the publication of this document:

Amduus Information Works, Inc.
1818 Briarwood
Flint, MI 48507
http://www.amduus.com

Article Submission Information:

Please submit your article in Microsoft Word format or as text. Please include a little bit about yourself for the About the Author paragraph.

Looking for technical articles, *marketing Progress* articles, articles about books relevant to programming/software industry, white papers, etc.

I am looking for work, if you have any knowledge of potential work, I would appreciate hearing from you!

http://www.amduus.com/Resumes/ScottAuge.html

Thanks!